

日 本 国 特 許 庁  
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office

出 願 年 月 日

Date of Application:

2002年12月18日

出 願 番 号

Application Number:

特願2002-366345

[ST.10/C]:

[JP2002-366345]

出 願 人

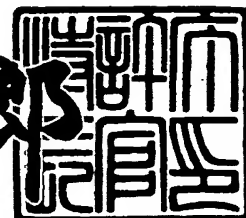
Applicant(s):

インターナショナル・ビジネス・マシーンズ・コーポレーション

2003年 6月 9日

特 許 庁 長 官  
Commissioner,  
Japan Patent Office

太田信一郎



出証番号 出証特2003-3044827

【書類名】 特許願

【整理番号】 JP9020212

【あて先】 特許庁長官殿

【国際特許分類】 G06F 11/28

【発明者】

    【住所又は居所】 神奈川県大和市下鶴間 1 6 2 3 番地 1 4 日本アイ・ピー・エム株式会社 大和事業所内

    【氏名】 稲垣 巖

【発明者】

    【住所又は居所】 神奈川県大和市下鶴間 1 6 2 3 番地 1 4 日本アイ・ピー・エム株式会社 大和事業所内

    【氏名】 古澤 修

【特許出願人】

    【識別番号】 390009531

    【氏名又は名称】 インターナショナル・ビジネス・マシーンズ・コーポレーション

【代理人】

    【識別番号】 100086243

    【弁理士】

    【氏名又は名称】 坂口 博

【代理人】

    【識別番号】 100091568

    【弁理士】

    【氏名又は名称】 市位 嘉宏

【代理人】

    【識別番号】 100108501

    【弁理士】

    【氏名又は名称】 上野 剛史

【復代理人】

【識別番号】 100085408

【弁理士】

【氏名又は名称】 山崎 隆

【手数料の表示】

【予納台帳番号】 117560

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 9706050

【包括委任状番号】 9704733

【包括委任状番号】 0207860

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 情報処理装置、トレース処理方法、プログラム及び記録媒体

【特許請求の範囲】

【請求項 1】 処理状況に関する情報を収集するトレース処理が含まれるルーチンをマルチスレッドで実行可能な情報処理装置において、

起動中のスレッド毎に、実行中のルーチンの登録を行うルーチン登録手段と、  
実行中のルーチンにおけるトレース処理のレベルを前記登録情報に基づいて判定するレベル判定手段とを具備することを特徴とする情報処理装置。

【請求項 2】 前記レベル判定手段は、前記実行中のルーチンにおけるトレース処理のレベル判定を、そのルーチンと同一又は所定の関係にあるルーチンの登録がなされているか否かに基づいて行うものであることを特徴とする請求項 1 に記載の情報処理装置。

【請求項 3】 前記所定の関係を相互に有するルーチンを規定するテーブルを有することを特徴とする請求項 2 に記載の情報処理装置。

【請求項 4】 前記テーブルの設定又は変更を行うための入力を受け入れ、前記テーブルの設定又は変更を行う手段を有することを特徴とする請求項 3 に記載の情報処理装置。

【請求項 5】 前記所定の関係にあるルーチンは、共有のリソースにアクセスする関係にあるルーチンであることを特徴とする請求項 2 に記載の情報処理装置。

【請求項 6】 前記レベル判定手段は、前記実行中のルーチンにおけるトレース処理のレベル判定に際し、そのルーチンと同一又は所定の関係にあるルーチンの登録がなされている場合、トレース処理のレベルはトレース処理の実行が制限なく許容されるレベルである旨の判定を行うものであることを特徴とする請求項 1 に記載の情報処理装置。

【請求項 7】 前記レベル判定手段は、前記実行中のルーチンにおけるトレース処理のレベル判定に際し、そのルーチンと同一又は所定の関係にあるルーチンの登録がなされていない場合、トレース処理のレベルはトレース処理の実行がまったく許容されないレベルである旨の判定を行うものであることを特徴とする

請求項 1 に記載の情報処理装置。

【請求項 8】 前記レベル判定手段は、前記登録が行われる際に、登録対象ルーチンと同一又は所定の関係にあるルーチンの登録がなされている場合、その旨を示す同時実行情報を、双方の登録情報に対して付加するものであることを特徴とする請求項 1 に記載の情報処理装置。

【請求項 9】 前記レベル判定手段は、前記同時実行情報の付加を行う際に、前記双方の登録情報に対し、前記トレース処理のレベルを示すトレース・レベル情報を付加するものであることを特徴とする請求項 8 に記載の情報処理装置。

【請求項 10】 前記トレース処理が含まれるルーチンを含むサーバ・プログラムと、クライアント・プログラムからの前記ルーチンの呼び出しに応じて前記ルーチンの実行を行うアプリケーション実行制御部と、前記アプリケーション実行制御部からの依頼に応じてトレース処理を実行するトレース実行部とを備え、前記トレース実行部はトレース処理の実行に際し、そのトレース処理のレベルを前記レベル判定手段に問い合わせるものであることを特徴とする請求項 1 に記載の情報処理装置。

【請求項 11】 前記トレース処理が含まれるルーチンを使用するプログラムと、前記トレース処理が含まれるルーチンが格納された共有ライブラリと、前記プログラムで使用されるルーチンからの依頼に応じてトレース処理を実行するトレース実行部とを備え、前記トレース実行部はトレース処理の実行に際し、そのトレース処理のレベルを前記レベル判定手段に問い合わせるものであることを特徴とする請求項 1 に記載の情報処理装置。

【請求項 12】 処理状況に関する情報を収集するトレース処理が含まれるルーチンをマルチスレッドで実行可能な情報処理装置におけるトレース処理方法であって、

起動中のスレッド毎に、実行中のルーチンの登録を前記情報処理装置が行うルーチン登録工程と、

実行中のルーチンにおけるトレース処理のレベルを前記登録情報に基づいて前記情報処理装置が判定するレベル判定工程とを具備することを特徴とするトレース処理方法。

【請求項 1 3】 前記レベル判定工程では、前記実行中のルーチンにおけるトレース処理のレベル判定を、そのルーチンと同一又は所定の関係にあるルーチンの登録がなされているか否かに基づいて行うことを特徴とする請求項 1 2 に記載のトレース処理方法。

【請求項 1 4】 前記トレースのレベル判定に際し、前記所定の関係を相互に有するルーチンを規定するテーブルを参照することを特徴とする請求項 1 3 に記載のトレース処理方法。

【請求項 1 5】 前記情報処理装置が、前記テーブルの設定又は変更を行うための入力を受け入れ、前記テーブルの設定又は変更を行う工程を有することを特徴とする請求項 1 4 に記載のトレース処理方法。

【請求項 1 6】 前記所定の関係にあるルーチンは、共有のリソースにアクセスする関係にあるルーチンであることを特徴とする請求項 1 3 に記載のトレース処理方法。

【請求項 1 7】 前記レベル判定工程では、前記実行中のルーチンにおけるトレース処理のレベル判定に際し、そのルーチンと同一又は所定の関係にあるルーチンの登録がなされている場合、トレース処理のレベルはトレース処理の実行が制限なく許容されるレベルである旨の判定を行うことを特徴とする請求項 1 2 に記載のトレース処理方法。

【請求項 1 8】 前記レベル判定工程では、前記実行中のルーチンにおけるトレース処理のレベル判定に際し、そのルーチンと同一又は所定の関係にあるルーチンの登録がなされていない場合、トレース処理のレベルはトレース処理の実行がまったく許容されないレベルである旨の判定を行うものであることを特徴とする請求項 1 2 に記載のトレース処理方法。

【請求項 1 9】 前記レベル判定工程では、前記登録が行われる際に、登録対象ルーチンと同一又は所定の関係にあるルーチンの登録がなされている場合、その旨を示す同時実行情報を、双方の登録情報に対して付加することを特徴とする請求項 1 2 に記載のトレース処理方法。

【請求項 2 0】 前記レベル判定工程では、前記同時実行情報の付加を行う際に、前記双方の登録情報に対し、前記トレース処理のレベルを示すトレース・

レベル情報を付加することを特徴とする請求項 1 9 に記載のトレース処理方法。

【請求項 2 1】 前記トレース処理が含まれるルーチンを含むサーバ・プログラムを前記情報処理装置が実行する工程と、クライアント・プログラムからの呼び出しに応じて前記ルーチンの実行を前記情報処理装置が行うアプリケーション実行制御工程と、前記アプリケーション実行制御工程で発生したトレース処理を前記情報処理装置が実行するトレース実行工程とを備え、前記トレース実行工程ではトレース処理の実行に際し、前記レベル判定工程における判定結果を参照することを特徴とする請求項 1 2 に記載のトレース処理方法。

【請求項 2 2】 前記トレース処理が含まれるルーチンを使用するプログラムを前記情報処理装置が実行する工程と、共有ライブラリに格納された前記トレース処理が含まれるルーチンを前記情報処理装置が実行する工程と、前記ルーチン実行工程におけるトレース処理を前記情報処理装置が実行するトレース実行工程とを備え、前記トレース実行工程ではトレース処理の実行に際し、前記レベル判定工程における判定結果を参照することを特徴とする請求項 1 2 に記載のトレース処理方法。

【請求項 2 3】 コンピュータを請求項 1 ～ 1 1 のいずれかの情報処理装置及びそれを構成する各手段として機能させることを特徴とするプログラム。

【請求項 2 4】 コンピュータを、  
該コンピュータにおいて起動中のスレッドで実行中のルーチンを登録する旨の依頼に応じ、スレッド毎にその登録を行うルーチン登録手段、及び  
実行中のルーチンにおけるトレース処理のレベルの問合せに応じ、前記登録情報に基づき、前記問合せに対する回答を行う回答手段として機能させることを特徴とするプログラム。

【請求項 2 5】 請求項 2 3 又は 2 4 のプログラムを記録したことを特徴とするコンピュータ読取り可能な記録媒体。

【発明の詳細な説明】

【0 0 0 1】

【発明の属する技術分野】

本発明は、トレース処理が含まれるルーチンをマルチスレッドで実行可能な情

報処理装置、この情報処理装置におけるトレース処理方法、コンピュータを前記情報処理装置として機能させるプログラム、及びこのプログラムを記録したコンピュータ読取り可能な記録媒体に関する。

#### 【0002】

##### 【従来の技術】

従来、プログラムに障害が発生した場合にその原因を解析し、特定するため、プログラムの実行状況を標準出力やファイル等に時系列で出力するトレースという手法が知られている。しかしながら、トレースを用いてより多くの情報を取得しようとする、本来の業務ロジックの処理以外に、プログラムの実行状況を記録するための余分な処理が必要となるため、プログラム全体としてのパフォーマンスの低下を招く。また、適切な範囲で情報の収集を行わないと、大量のデータが出力されることになり、システム・リソースを圧迫する要因となりうる。

#### 【0003】

そこで、トレースを必要最小限の範囲で実施することによって、パフォーマンスの低下やシステム・リソースの圧迫を抑えることを目的として従来、次のような工夫が行われている。たとえば、収集すべき情報を重要度に応じて分類し、必要とする情報の重要度をユーザが指定することによって重要度の低い情報が取得されるのを防止するものや、特定のスレッドに対するトレース機能を設け、特定のスレッド上で実行される処理についてのみ情報の収集を行うものが知られている。

#### 【0004】

また、スレッドを排他実行スレッドと、他のスレッドとに分類し、排他実行スレッドについては同時実行させないようにしてスレッドの実行順序を制御しつつ情報収集を行うことにより、デバッグ効率の向上を図ったデバッグ装置も知られている（たとえば特許文献1参照）。また、通常のインタプリタと、トレース用のインタプリタとを設け、これらを切り替えて使用することにより、トレース機能の実行可否を制御できるようにした技術も知られている（たとえば特許文献2参照）。

#### 【0005】



【特許文献 1】

特開平 1 1 - 3 3 8 7 3 3 号公報

【特許文献 2】

特開 2 0 0 2 - 5 5 8 4 8 号公報

【0 0 0 6】

【発明が解決しようとする課題】

しかしながら、これら従来のトレース技術によれば、プログラムの実行前又は実行中にユーザによってなされる設定又は設定変更に従って予め特定されるスレッドやプログラム部分等についてのみトレースの可否を制御し得るだけである。つまり、実際の運用下における種々の実行要求に応じて動的に変化するスレッドの生成状況に対応した情報収集を行うことができない。したがって、従来のトレース技術をマルチスレッド環境下でのみ発生する障害の分析に適用すると、障害の発生条件に合致しない場合、たとえばシングル・スレッドでプログラムが実行されるような場合においてもトレース処理による情報の収集が行われ、収集された情報には障害の発生原因とは無関係のデータが多く含まれることになる。このためパフォーマンスの低下やシステム・リソースの圧迫に加え、トレース出力の分析に大きな労力が必要となる。

【0 0 0 7】

また、マルチスレッド・プログラムにおいて、複数のスレッドが同時に特定の処理を実行することが原因となって発生する障害は、各スレッドの実行状況などのタイミングに依存する。このため、単一のスレッドで発生する障害に比べて問題の原因を特定することが困難である。

【0 0 0 8】

さらに、マルチスレッド環境下でのみ発生する障害を分析する目的でトレースを使用する場合、従来のトレース技術によれば、収集する情報には障害の発生原因とは無関係のデータが多く含まれ、多くの余分な処理が発生するので、実際の運用環境においてはトレースの取得が許されない場合がある。

【0 0 0 9】

本発明の目的は、このような従来技術の問題点に鑑み、複数のスレッドにおい

て共通の資源をアクセスすることが原因で発生する問題の解析を、プログラムの実行環境に大きな影響を与えることなく、容易に行うことができるようなトレース技術を提供することにある。

#### 【0010】

##### 【課題を解決するための手段】

この目的を達成するため、本発明に係るトレース処理方法は、処理状況に関する情報を収集するトレース処理が含まれるルーチンをマルチスレッドで実行可能な情報処理装置におけるトレース処理方法であって、起動中のスレッド毎に、実行中のルーチンの登録を前記情報処理装置が行うルーチン登録工程と、実行中のルーチンにおけるトレース処理のレベルを前記登録情報に基づいて前記情報処理装置が判定するレベル判定工程とを具備することを特徴とする。

#### 【0011】

ここで、トレース処理としては、たとえば所定のメッセージや所定の変数の内容をログ・ファイル等に出力する処理が該当する。ルーチンとは何らかの処理を行う一連のプログラムコードであり、たとえば、関数やメソッドが該当する。トレース処理のレベルの判定としては、たとえばトレース処理を行うか否かの判定や、トレース処理を収集可能なすべての情報について行うか、一部の情報については省略して行うか、又はまったく行わないかの判定が該当する。実行中のルーチンの登録とは、実行開始時に登録を行い、実行終了時に登録を抹消することを意味する。

#### 【0012】

この構成において、ルーチン登録工程では、ルーチンの実行のためにスレッドが起動するときにそのルーチンの登録を行い、実行が終了してスレッドが消滅するときにルーチンの登録を抹消する。これにより、実行中のルーチンの登録がスレッド毎に行われる。この実行中のルーチンの登録情報に基づき、レベル判定工程では、実行中のルーチンにおけるトレース処理のレベルを判定する。その際、たとえば、同一ルーチンが異なるスレッドで実行中であればトレース処理は許容され、さもないと許容されないレベルであると判定することができるが、このような判定結果は、登録情報の経時的変化に応じて刻々変化する。つまり、レベ

ル判定工程では、実行中のルーチンにおけるトレース処理の可否や程度が、どのようなルーチンが他のスレッドで実行中であるかに応じ、動的に判定されることになる。この判定結果に基づき、トレース処理の可否やレベルを動的に制御することができる。

#### 【0013】

本発明に係るトレース処理方法において、前記レベル判定工程では、実行中のルーチンにおけるトレース処理のレベル判定を、そのルーチンと同一又は所定の関係にあるルーチンの登録がなされているか否かに基づいて行うことができる。この場合、前記所定の関係を相互に有するルーチンを規定するテーブルを参照するようにしてもよい。さらに、情報処理装置がこのテーブルの設定又は変更を行うための入力を受け入れ、テーブルの設定又は変更を行うようにしてもよい。前記所定の関係にあるルーチンとしてはたとえば、共有のリソースにアクセスする関係にあるルーチンや、同時に実行すると、何らかの原因により問題が生じ得る関係にあるルーチンが該当する。

#### 【0014】

また、本発明に係るトレース処理方法において、レベル判定工程としてはたとえば、実行中のルーチンにおけるトレース処理のレベル判定に際し、そのルーチンと同一又は所定の関係にあるルーチンの登録がなされており又はなされていない場合、トレース処理のレベルはトレース処理の実行が制限なく許容され又はまったく許容されないレベルである旨の判定を行うものが該当する。あるいは、ルーチンの登録が行われる際に、登録対象ルーチンと同一又は所定の関係にあるルーチンの登録がなされている場合、その旨を示す同時実行情報を、双方の登録情報に対して付加するものが該当する。同時実行情報の付加を行う際に、双方の登録情報に対し、トレース処理のレベルを示すトレース・レベル情報を付加するようにしてもよい。

#### 【0015】

また、本発明に係るトレース処理方法は、トレース処理が含まれるルーチンを含むサーバ・プログラムを情報処理装置が実行する工程と、クライアント・プログラムからの呼び出しに応じて前記ルーチンの実行を情報処理装置が行うアプリ

ケーション実行制御工程と、アプリケーション実行制御工程で発生したトレース処理を情報処理装置が実行するトレース実行工程とを備えてもよい。あるいは、トレース処理が含まれるルーチンを使用するプログラムを情報処理装置が実行する工程と、共有ライブラリに格納された前記トレース処理が含まれるルーチンを情報処理装置が実行する工程と、前記ルーチン実施工程におけるトレース処理を情報処理装置が実行するトレース実行工程とを備えてもよい。これらの場合、トレース実行工程では、トレース処理の実行に際し、レベル判定工程における判定結果を参照することができる。

## 【 0 0 1 6 】

本発明に係る情報処理装置は、上述の本発明に係るトレース処理方法を構成する各工程を行う手段によって構成される。

## 【 0 0 1 7 】

本発明に係るプログラムは、本発明に係る情報処理装置及びそれを構成する各手段としてコンピュータを機能させるものであることを特徴とする。

## 【 0 0 1 8 】

本発明に係る記録媒体は、本発明に係るプログラムを記録したコンピュータ読取り可能な記録媒体であることを特徴とする。

## 【 0 0 1 9 】

## 【発明の実施の形態】

## 【 0 0 2 0 】

図 1 は本発明を適用し得る O R B (Object Request Broker) による通信基盤を基礎とするクライアント・サーバ・システムを示す。O R B はクライアントとサーバ・オブジェクトとの間の接続を確立し、クライアントがサーバ・オブジェクトのメソッドを呼び出すことを可能にするソフトウェアである。このシステムにおいて稼動するクライアント・サーバ・アプリケーションはサーバ・プログラム 1 及びクライアント・マシン 2 におけるクライアント・プログラムを有する。サーバ・プログラム 1 が定義するオブジェクトのメソッドはアプリケーション実行制御部 3 により実行される。サーバ・プログラム 1 ではたとえばクラス A に属するインスタンス a 及び b が定義されており、各インスタンスはトレース処理

が含まれたメソッドX及びメソッドYを備えている。アプリケーション実行制御部3は、たとえばJava（登録商標）仮想マシンに相当するものであり、プラットフォームに依存しないバイトコードを実行することによって、サーバのOS（オペレーティング・システム）の如何に関係なく、メソッドの実行を可能にしている。

## 【0021】

クライアント・プログラムがメソッドXを呼び出すと、アプリケーション実行制御部3は、OSが生成する単一のスレッドにおいて、メソッドXを構成するバイトコードを順次解釈して実行する。異なるクライアント・マシーン2における各クライアント・プログラムがメソッドXを同時に呼び出すと、異なるアプリケーション実行制御部3が、クラスAのインスタンスa及びbのメソッドXを、異なるスレッドにおいて並行して実行する。

## 【0022】

このようなシステムにおいては、従来のトレース手法によれば、グローバル変数等の静的に設定されたトレース制御情報を参照し、全スレッドにおいてそれぞれトレース処理の必要性の有無を判断していた。これに対し、本発明の第1の実施形態においては図2に示すように、図1のシステムに対し、トレース制御部（Trace Manager）21及びスレッド処理多重度管理テーブル22を導入し、トレース実行部（Trace Executioner）23によるログ・ファイル24へのトレース情報の出力を、起動中のスレッドで実行されているメソッドが同一であるか否かに応じ、動的に制御するようにしている。

## 【0023】

スレッド処理多重度管理テーブル22は、起動中のスレッド毎に、実行中のメソッドを登録するためのテーブルである。テーブルを構成する各レコードは、起動中のスレッドを識別するためのスレッド識別子、対応スレッドで実行中であるメソッドを特定するクラス名及びメソッド名、並びにトレース・レベルの各項目を有する。トレース・レベルの項目には、対応スレッドで実行中のメソッドにおけるトレース処理の実行可否を示すフラグがセットされる。フラグとしては、トレース処理の実行を許容する旨を示す"5"及びトレース処理の実行を禁止する旨

を示す”0”が用いられる。

【0024】

トレース制御部21は、アプリケーション実行制御部3からの要求に応じ、生成されたスレッドの管理テーブル22への登録を行うメソッド”addThreadInfo()”、及び消去されたスレッドの登録を管理テーブル22から抹消するメソッド”removeThreadInfo()”を備えている。またトレース実行部23からの、スレッド識別子を特定した問合せに応じ、そのスレッドにおけるトレース処理の可否について、管理テーブル22を参照してチェックし、回答を行うメソッド”checkTrace()”を備える。トレース実行部23はアプリケーション実行制御部3の要求に応じ、トレース処理の可否をトレース制御部21に問い合わせ、回答結果に応じてトレース処理を実行し、又は省略するメソッド”setLogEvent()”を備える。

【0025】

この構成において、クライアント・プログラムがサーバ・プログラム1のメソッドを呼び出すと、サーバではOSによりスレッドが生成され、対応するアプリケーション実行制御部3に対し、呼び出されたメソッドの実行要求がなされる（図中の①の処理）。

【0026】

この実行要求を受けると、アプリケーション実行制御部3は、呼び出されたメソッドを実際に実行する前に、トレース制御部21に対し、生成されたスレッドに関してスレッド処理多重度管理テーブル22への登録を要求するとともに、生成されたスレッドの識別子、呼び出されたメソッドのクラス名及びメソッド名を通知する（Call addThreadInfo()；処理②）。

【0027】

この要求に応じ、トレース制御部21は、通知されたスレッドの識別子並びにクラス名及びメソッド名をスレッド処理多重度管理テーブル22に登録する（addThreadInfo()；処理③）。その際、トレース制御部21は、登録するメソッドとクラス名及びメソッド名が同一であるメソッドが管理テーブル22に登録済みであるかどうかを調べ、登録済みであることが判明した場合は、それら同一メソ

ッドの登録レコードにおけるトレース・レベルを"5"に設定する(処理④)。

【0028】

たとえば、スレッド処理多重度管理テーブル22が図2に示すような状態にあり、登録されるスレッドの識別子が"500"、メソッドのクラス名及びメソッド名が"class A"及び"Method X"であるとすれば、これと同一のクラス名及びメソッド名のメソッドが、スレッド識別子が"100"のものとして登録済みであるため、スレッド識別子が"500"及び"100"のレコードにおけるトレース・レベルが"5"にセットされることになる。

【0029】

一方、アプリケーション実行制御部3は、クライアント・プログラムが呼び出したメソッドを実行しているときにトレース処理たとえばプリント命令"Print"Trace2""に遭遇すると、そのトレース処理をトレース実行部23に依頼する(Print"Trace2"; 処理⑤)。この依頼を受けると、トレース実行部23は、トレース制御部21に対し、依頼されたトレース処理の実行の可否について問合せを行う(処理⑥)。この問合せには、依頼されたトレース処理に係るスレッドの識別子を伴う。

【0030】

この問合せに応じ、トレース制御部21は、スレッド処理多重度管理テーブル22中の問合せに係るスレッド識別子のレコードにおけるトレース・レベルの値を調べ(checkTrace(); 処理⑦)、トレース処理の可否をトレース実行部23に通知する(処理⑥)。この通知に基づきトレース実行部23は、依頼されたトレース処理を実行し(setLogEvent(); 処理⑧)、又は省略する。たとえば、依頼されたトレース処理が、識別子が"100"のスレッドにおけるトレース処理であるとすれば、スレッド処理多重度管理テーブル22における該当レコードのトレース・レベルは"5"であるため、トレース処理が実行されることになる。

【0031】

他方、クライアント・プログラムから呼び出されているいずれかのメソッドの実行が終了し、対応するスレッドが消去されるとき、対応するアプリケーション実行制御部3は、消去されるスレッドに関する登録の抹消をトレース制御部21に

依頼する (Call removeThreadInfo() ; 処理⑨) 。この依頼を受けると、トレース制御部 2 1 は、登録抹消を依頼されたスレッドに関する登録をスレッド処理多重度管理テーブル 2 2 から抹消する (removeThreadInfo() ; 処理(10)) 。その際、削除するレコードとクラス名及びメソッド名が同一であるレコードについては、さらにクラス名及びメソッド名が同一である他のレコードが存在しない限り、トレース・レベルを"0"とする。たとえば、図 2 の管理テーブル 2 2 の状態においてスレッド識別子が"1 0 0"であるレコードが削除されるとき、そのクラス名" c l a s s    A"及びメソッド名" M e t h o d    X"が同一であるスレッド識別子が"5 0 0"のレコードのトレース・レベルは"0"に変更されることになる。

#### 【 0 0 3 2 】

図 3 ～図 5 は上述のトレース制御部 2 1 の処理を示す。トレース制御部 2 1 はアプリケーション実行制御部 3 からの生成されたスレッドに関する登録依頼又は消去されるスレッドに関する登録抹消依頼を受けた場合は図 3 の登録処理 (addThreadInfo()) 又は図 4 の登録抹消処理 (removeThreadInfo()) を行い、トレース実行部 2 3 からのトレース処理の可否に関する問合せがあった場合には図 5 の実行可否チェック処理 (checkTrace()) を行う。

#### 【 0 0 3 3 】

登録に際しては、図 3 に示すように、スレッド処理多重度管理テーブル 2 2 に対し、図 2 に示される形式の 1 レコードを追加するが (ステップ 4 1) 、その際、同一メソッドに関する登録が既に行われている場合は双方のレコードにおけるトレース・レベルを"5"にセットし、さもなければ追加レコードにおけるトレース・レベルを"0"にセットする (ステップ 4 2 ～4 4) 。登録抹消に際しては図 4 に示すように、対応レコードを削除するとともに (ステップ 5 1) 、削除レコードと同一のメソッドに関するレコードが存在し、かつそのようなレコードが 1 つのみである場合はそのトレース・レベルを"0"にセットする (ステップ 5 2、5 3) 。トレース処理の実行可否のチェックに際しては、図 5 に示すように、そのトレース処理に係るスレッドの識別子をキーにして対応レコードのトレース・レベルをチェックし、その結果をトレース実行部 2 3 に通知する (ステップ 6 1、6 2) 。



## 【 0 0 3 4 】

このようにしてトレース制御部 2 1 は常に、実行中のスレッドのみがすべて登録され、かつクラス名及びメソッド名が同一であるスレッドについては、トレース・レベルが” 5 ”となるようにスレッド処理多重度管理テーブル 2 2 の登録内容を維持し、管理するとともに、スレッド処理多重度管理テーブル 2 2 に基づいて、各スレッドにおけるトレース処理の実行可否のチェックを行う。

## 【 0 0 3 5 】

本実施形態によれば、トレース処理が含まれるメソッドをマルチスレッドで実行可能なサーバにおいて、起動中のスレッド毎に、実行中のメソッドの登録を行い、実行中のメソッドにおけるトレース処理の可否を登録情報に基づいて判定するようにしたため、判定時点で起動中の各スレッドにおいて実行されているメソッドに応じ、動的にトレース処理の可否の判定を行うことができる。つまり、共有リソースにアクセスすることによる問題が生じる可能性は、どのようなメソッドが同時に実行されているかに応じて刻々変化するが、その変化する状況に応じて、問題解決のためのトレース処理を、問題が生じる可能性がある場合に限定して行うことができる。したがって、プログラムの実行環境に大きな影響を与えることなく、不要な情報の収集を排除したトレース処理を行い、問題解決のための解析の容易化を図ることができる。

## 【 0 0 3 6 】

また、実行中のメソッドにおけるトレース処理の可否の判定を、そのメソッドと同一の関係にあるメソッドの登録がなされているか否かに基づいて行うようにしたため、共有リソースにアクセスすることによる問題が生じる可能性が高い場合に限定してトレース処理を行うことができる。

## 【 0 0 3 7 】

また、スレッド処理多重度管理テーブル 2 2 への登録が行われる際に、登録対象メソッドと同一の関係にあるメソッドの登録がなされている場合、その旨を示す同時実行情報を” 5 ”として、双方の登録レコードにおけるトレース・レベルの項目に記録するようにしたため、トレース処理の実行可否の判定結果が必要な時には、スレッド処理多重度管理テーブル 2 2 を再度探索することなく直ちに判定

結果を得ることができる。

【0038】

以上の実施形態では最も単純な条件、すなわち同一メソッドの同時実行が開始された時点でトレース処理の実行可否の条件を更新するようにしているが、次に示す第2の実施形態では、より複雑な条件でトレース処理実行可否の制御を行うようにしている。すなわちトレース条件管理テーブルを導入し、このテーブルにおいてトレース処理の実行可否の条件をより詳細に規定することにより、トレース処理実行可否の制御をより柔軟に行うことを可能にしている。

【0039】

図6は本発明の第2の実施形態に係るシステムの主要部、及び処理の流れを示す。このシステムでは、第1実施形態におけるトレース制御部21の代わりにトレース制御部31を備え、スレッド処理多重度管理テーブル22のトレース・レベルを、トレース条件管理テーブル32を参照して設定又は変更するようにしている。また、トレース・レベルとして、トレース処理の許容又は省略の2つのレベルのみではなく、より多くのレベルを設定できるようにしている。この設定レベルに応じて、トレース処理可否の問合せに対する回答が行われることになる。他の点については上述第1実施形態の場合と同様である。

【0040】

トレース条件管理テーブル32は複数のトレース・パターンを定義している。テーブル中の各レコードはトレース・パターン、トレース・レベル、及びクラス／メソッドの各項目を有する。トレース・パターンは各トレース・パターンを区別するためのパターン番号、トレース・レベルの項目には対応トレース・パターンにおけるトレース処理の程度を示すレベル番号、クラス／メソッドの項目には対応トレース・パターンが適用されるメソッドを特定するクラス名及びメソッド名が格納される。

【0041】

各レコードにおけるクラス／メソッドの項目には特定の関係を有する複数のメソッドのクラス名及びメソッド名を記録することができる。この特定関係を有するメソッド同士については、同時に実行されるとき、それらのメソッドにおける

トレース処理について、それらのメソッドを前記特定関係を有するものとして規定しているレコードのトレース・レベルが適用される。また、同一のメソッドが同時に実行されているときには、そのメソッドについて規定されているトレース・レベルが適用される。

#### 【0042】

前記特定関係を有するメソッドとしては、たとえば共通のリソースにアクセスするメソッドが該当する。図6の例では、トレース・パターン”2”のレコードにおいて、”class A::Method Y”及び”class B::Method Z”が前記特定関係を有するメソッドとして規定されている。

#### 【0043】

図7～図9はそれぞれ、トレース制御部31による登録処理、登録抹消処理、及び実行可否チェック処理を示す。トレース制御部31はアプリケーション実行制御部3（図2）からの生成されたスレッドに関する登録依頼又は消去されるスレッドに関する登録抹消依頼を受けた場合は図7の登録処理（addThreadInfo()）又は図8の登録抹消処理（removeThreadInfo()）を行い、トレース実行部23からのトレース処理の可否に関する問合せがあった場合には図9の実行可否チェック処理（checkTrace()）を行う。

#### 【0044】

登録処理においては、図7に示すように、まず、そのスレッドが実行するメソッドと同一のメソッドがクラス／メソッドの項目に記録されているレコードをトレース条件管理テーブル32においてサーチし、対応するトレース・レベルを取得する（ステップ81）。また、当該レコード中に前記特定関係にある他のメソッドのクラス名及びメソッド名が記録されていればそれも取得する（ステップ82）（図6中の処理①）。

#### 【0045】

次に、登録依頼を受けたスレッドの識別子、並びにそのスレッドが実行するメソッドのクラス名及びメソッド名を記録したレコードを追加することにより、スレッド処理多重度管理テーブル22に対して新たなエントリを加える（ステップ83）。このとき、追加レコードと同一のクラス名及びメソッド名を有するレコ

ードがスレッド処理多重度管理テーブル 2 2 中に存在するかどうかを判定し（ステップ 8 4）、存在する場合には、そのレコード及び追加レコード双方のトレース・レベルを前記取得したトレース・レベルに設定する（ステップ 8 6）。また前記取得したクラス名及びメソッド名と同一のものがスレッド処理多重度管理テーブル 2 2 中に存在するかどうかを判定し（ステップ 8 4）、存在する場合にはそのレコード及び追加レコード双方のトレース・レベルを、前記取得したトレース・レベルに設定する（ステップ 8 6）。同一又は取得したクラス名及びメソッド名と同一のものを有するレコードが存在しない場合は追加レコードのトレース・レベルを”0”に設定する（ステップ 8 7）。これにより、スレッドの登録が終了する。

## 【 0 0 4 6 】

一方、登録抹消処理においては、図 8 に示すように、まず、そのスレッドが実行するメソッドと前記特定関係を有するメソッドがトレース条件管理テーブル 3 2 において規定されているかどうかを調べ（ステップ 9 1）、特定関係を有するメソッドがあれば、そのクラス名及びメソッド名を取得する（ステップ 9 2）。

## 【 0 0 4 7 】

次に登録抹消依頼を受けたスレッドのレコードを削除する（ステップ 9 3）。このとき、削除レコードに係るメソッドと同一又は前記特定関係にあるメソッドのクラス名及びメソッド名を有するレコードがスレッド処理多重度管理テーブル 2 2 中に存在するか否かを判定し（ステップ 9 4）、存在する場合には、そのようなレコードがさらに他に存在する場合を除き、そのレコードのトレース・レベルを”0”に設定する（ステップ 9 5）。これにより、スレッドの登録抹消が終了する。

## 【 0 0 4 8 】

以上のスレッドの登録及び登録抹消の処理を通じて、スレッド処理多重度管理テーブル 2 2 には常に、起動中のスレッドのみが、実行中のメソッドを特定するクラス名及びメソッド名並びにトレース処理の可否及びレベルに関する情報とともに登録されることになる。

## 【 0 0 4 9 】

実行可否チェック処理においては、図9に示すように、問合せに係るスレッドに対応するトレース・レベルを、スレッド識別子をキーとしてスレッド処理多重度管理テーブル22から取得し（ステップ101）、トレース実行部23に送る（ステップ102）。

#### 【0050】

トレース実行部23は、送られたトレース・レベルに応じ、トレース処理を実行する。たとえば、トレース・レベルが"0"であればトレース処理を行わず、それ以外の値であれば、その値に応じて、トレース処理を部分的又は全体的に行ったり、一部省略したりする。

#### 【0051】

以上の登録及び登録抹消処理並びにトレース実行可否チェック処理の一例を具体的に示せば、次のようになる。すなわち、スレッド処理多重度管理テーブル22及びトレース条件管理テーブル32の内容が図6に示すような状態にある場合に、たとえば識別子が"400"であるスレッドの登録依頼があったとき、そのスレッドで実行されるメソッドのクラス/メソッド"`class A::Method Y`"はトレース条件管理テーブル32中に存在し、そのトレース・レベルは"4"である。また、そのレコード中には前記特定関係にある他のメソッド"`class B::Method Z`"が存在する。そして、この特定関係にあるメソッドはスレッド処理多重度管理テーブル22において識別子が"200"のスレッドに係るものとして既に登録されている。したがって、図7に示されるとおり、登録依頼された識別子"400"のスレッドがスレッド処理多重度管理テーブル22にエントリされるとともに、識別子"200"及び"400"の各レコードのトレース・レベルが"4"に設定される。

#### 【0052】

その後、識別子が"400"のスレッドについてトレース処理の可否の問合せがトレース実行部23（図2）からあったとすれば、スレッド処理多重度管理テーブル22中の当該レコードにおけるトレース・レベル"4"がトレース実行部23に通知される。トレース実行部23はこの通知されたトレース・レベルに応じてトレース処理を実行し、又は一部省略する。

## 【0053】

一方、アプリケーション実行制御部3から、識別子が”200”のスレッドについて登録抹消依頼があったとすれば、そのスレッドに係るクラス/メソッド”class B::Method Z”はトレース条件管理テーブル32においてトレース・パターン2のレコード中に存在する。また同じレコード中に、特定関係にあるメソッド/クラス”class A::Method Y”も定義されている。したがってこの場合、トレース制御部31は、スレッド処理多重度管理テーブル22における識別子”200”のレコードを削除するとともに、メソッドが特定関係にある識別子”400”のレコードにおけるトレース・レベルを”0”に設定する。以後、識別子”400”のスレッドにおけるトレース処理は禁止されることになる。

## 【0054】

図10は本発明の第3の実施形態に係るサーバにおけるシステム構成及び処理シーケンスを示す。この実施形態では、サーバ・プログラムが共有ライブラリ中のメソッドを呼び出すことによってマルチスレッドによる処理が行われる場合について示している。図10のシステムはサーバ・プログラム111、サーバ・プログラム111によって使用される共有ライブラリ112、実行中のスレッドのみを登録し、この登録情報に基づいてトレース処理の実行可否の問合せに回答するトレース制御部113、共有ライブラリ中の実行中のメソッドから依頼されるトレース処理の実行要求に応じ、トレース処理の実行可否をトレース制御部113に問い合わせ、その応答に基づいてトレース処理を実行し又は省略するトレース実行部114、及びスレッドの登録がなされるスレッド処理多重度管理テーブル115を備える。スレッド処理多重度管理テーブル115の各レコードには、登録されるスレッドの識別子、スレッドが実行するメソッドのメソッド名、及びトレース・レベルが記録される。トレース・レベルが”0”であれば対応スレッドにおけるトレース処理は禁止され、”5”であれば許容される。

## 【0055】

この構成において、サーバ・プログラム111が共有ライブラリ112中のメソッドたとえば”SQLExec( )”を呼び出すと(処理①)、メソッドを実行

するスレッドが生成される。すると、メソッドはまず、トレース制御部113に対し、生成されたスレッドの登録を依頼する（処理②）。この依頼は、図11の例ではトレース制御部113中の”addThreadInfo()”を呼び出すことによって行っている。

## 【0056】

この依頼を受けると、トレース制御部113は、依頼されたスレッドの登録を行う（addThreadInfo(); 処理③）。このときトレース制御部113は、スレッド処理多重度管理テーブル115において同一メソッドを実行しているスレッドが登録されているかどうかを調べ、登録されていなければトレース・レベルを”0”とし、登録されていれば、双方のトレース・レベルを”5”に設定する（処理④）。図10の例では、登録されるスレッド”500”が実行しているメソッド”SQLExec”は、スレッド”100”においても実行されているため、双方のトレース・レベルは”5”に設定されている。

## 【0057】

共有ライブラリ112中の実行中のメソッドが、トレース処理たとえば”Trace””Trace2”の実行をトレース実行部114に依頼すると（処理⑤）、トレース実行部114は、メソッドを実行しているスレッドの識別子を特定して、トレース制御部113に対し、トレース処理の実行可否を問い合わせる（処理⑥）。この問合せに応じ、トレース制御部113は、問合せに係るスレッドについて、スレッド処理多重度管理テーブル115におけるトレース・レベルに基づきトレース処理の実行可否を判定し（処理⑦）、その結果をトレース実行部に通知する（処理⑧）。図10の例ではこの処理をメソッド”checkTrace()”によって行っている。たとえば問合せに係るメソッドが”SQLExec”であれば、トレース・レベルが”5”であるため、トレース処理の実行を許容する旨を通知する。この通知に基づいてトレース実行制御部114は、依頼されたトレース処理の実行を、ログ・ファイル24を出力先として行う（setLogEvent(); 処理⑨）。

## 【0058】

共有ライブラリ112中の実行中のメソッドが終了するとき、メソッドはそれ

を実行しているスレッドの登録抹消をトレース制御部 1 1 3 に依頼する (Call removeThreadInfo(); 処理⑨)。この依頼に応じてトレース制御部 1 1 3 はそのスレッドの登録を抹消する (removeThreadInfo(); 処理(10))。その際、削除するレコードとメソッド名が同一であるレコードについては、さらにメソッド名が同一である他のレコードが存在しない限り、トレース・レベルを"0"とする。たとえば、図 1 0 のスレッド処理多重度管理テーブル 1 1 5 の状態においてスレッド識別子が"1 0 0"であるレコードが削除されるとき、そのメソッド名"SQL Exec"が同一であるスレッド識別子が"5 0 0"のレコードのトレース・レベルは"0"に変更される。

## 【0 0 5 9】

本実施形態におけるトレース制御部 1 1 3 の処理は、メソッドの識別をメソッド名のみによって行っている以外は、第 1 実施形態におけるトレース制御部 2 1 の場合とほぼ同じである。本実施形態によっても第 1 実施形態の場合と同様の効果を得ることができる。

## 【0 0 6 0】

なお、本発明は上述実施形態に限定されることなく適宜変形して実施することができる。たとえば、上述においてはスレッド処理多重度管理テーブル 2 2 においてスレッド識別子の項目が設けられているが、この項目は、スレッド毎にメソッドが登録されていれば、設けなくてもよい。ただし、テーブルのサーチ処理が煩雑になる可能性がある。また、上述においては言及しなかったが、トレース条件管理テーブル 3 2 の内容を設定し又は更新するための入力を受け入れ、その設定又は更新を行う手段を設けるようにしてもよい。

## 【0 0 6 1】

## 【発明の効果】

以上説明したように本発明によれば、起動中のスレッド毎に、実行中のルーチンの登録を行い、この登録情報に基づいて、実行中のルーチンにおけるトレース処理のレベルを判定するようにしたため、プログラムの実行環境に大きな影響を与えることなく、共有リソースを同時にアクセスすることによって生じる問題の解決に必要な情報の収集を行うことができる。



【図面の簡単な説明】

【図 1】

本発明を適用し得る O R B 通信基盤を基礎とするクライアント・サーバ・システムを示す図である。

【図 2】

本発明の一実施形態に係る O R B 通信基盤を基礎とするサーバにおけるシステム及び該システムにおける処理シーケンスを示す図である。

【図 3】

図 2 のシステムにおける登録処理を示すフローチャートである。

【図 4】

図 2 のシステムにおける登録抹消処理を示すフローチャートである。

【図 5】

図 2 のシステムにおける実行可否チェック処理を示すフローチャートである。

【図 6】

本発明の第 2 の実施形態に係るシステム及び処理の流れを示す図である。

【図 7】

図 6 のシステムのトレース制御部による登録処理を示すフローチャートである。

【図 8】

図 6 のシステムのトレース制御部による登録抹消処理を示すフローチャートである。

【図 9】

図 6 のシステムのトレース制御部による実行可否チェック処理を示すフローチャートである。

【図 1 0】

本発明の第 3 の実施形態に係るサーバにおけるシステム構成及び処理シーケンスを示す。

【符号の説明】

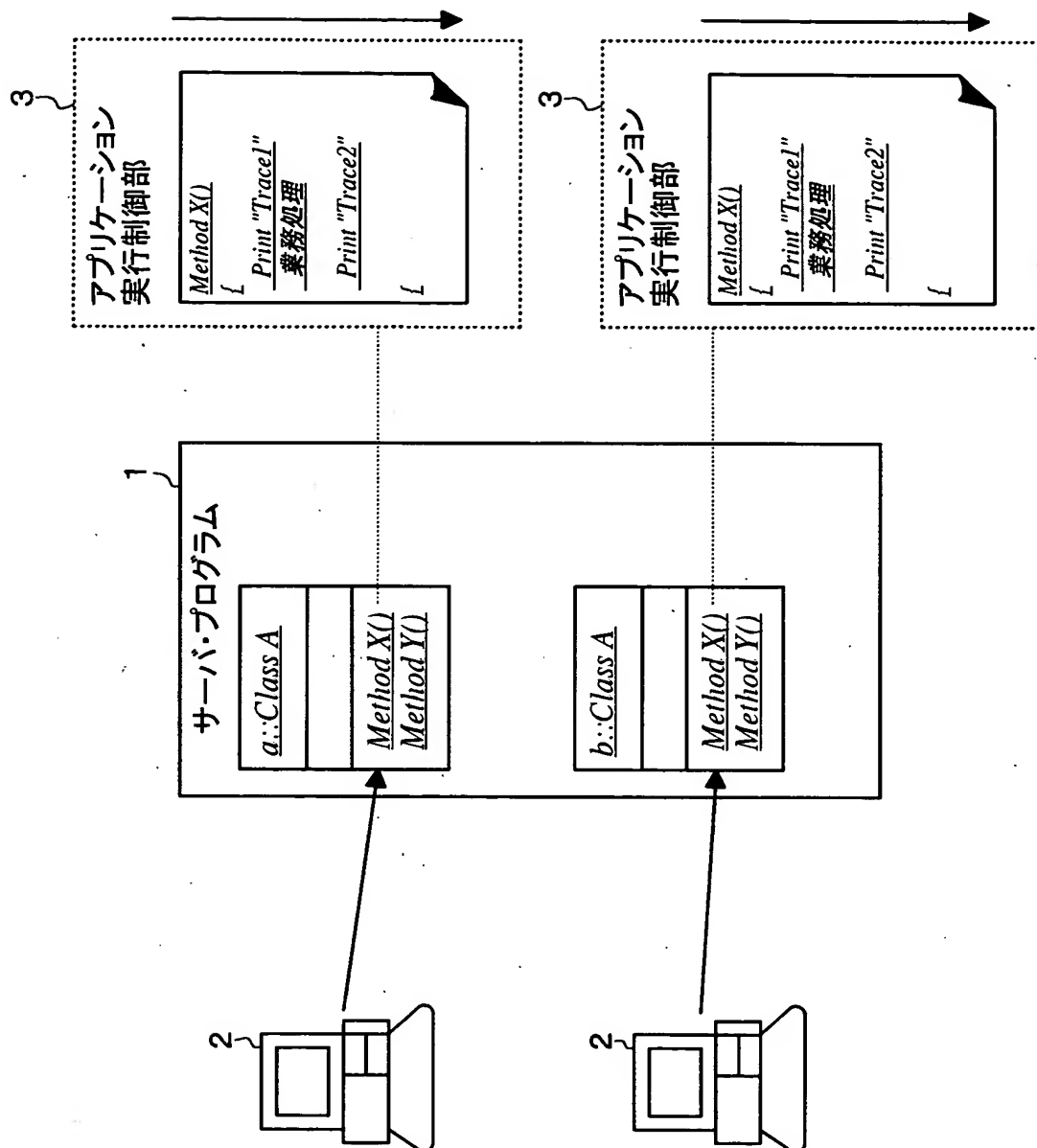
1, 1 1 1 : サーバ・プログラム、 2 : クライアント・マシン、 3 : アプリ

ケーション実行制御部、21, 31, 113 : トレース制御部、22, 115 :  
スレッド処理多重度管理テーブル、23 : トレース実行部、24 : ログ・ファイ  
ル、32 : トレース条件管理テーブル、112 : 共有ライブラリ。

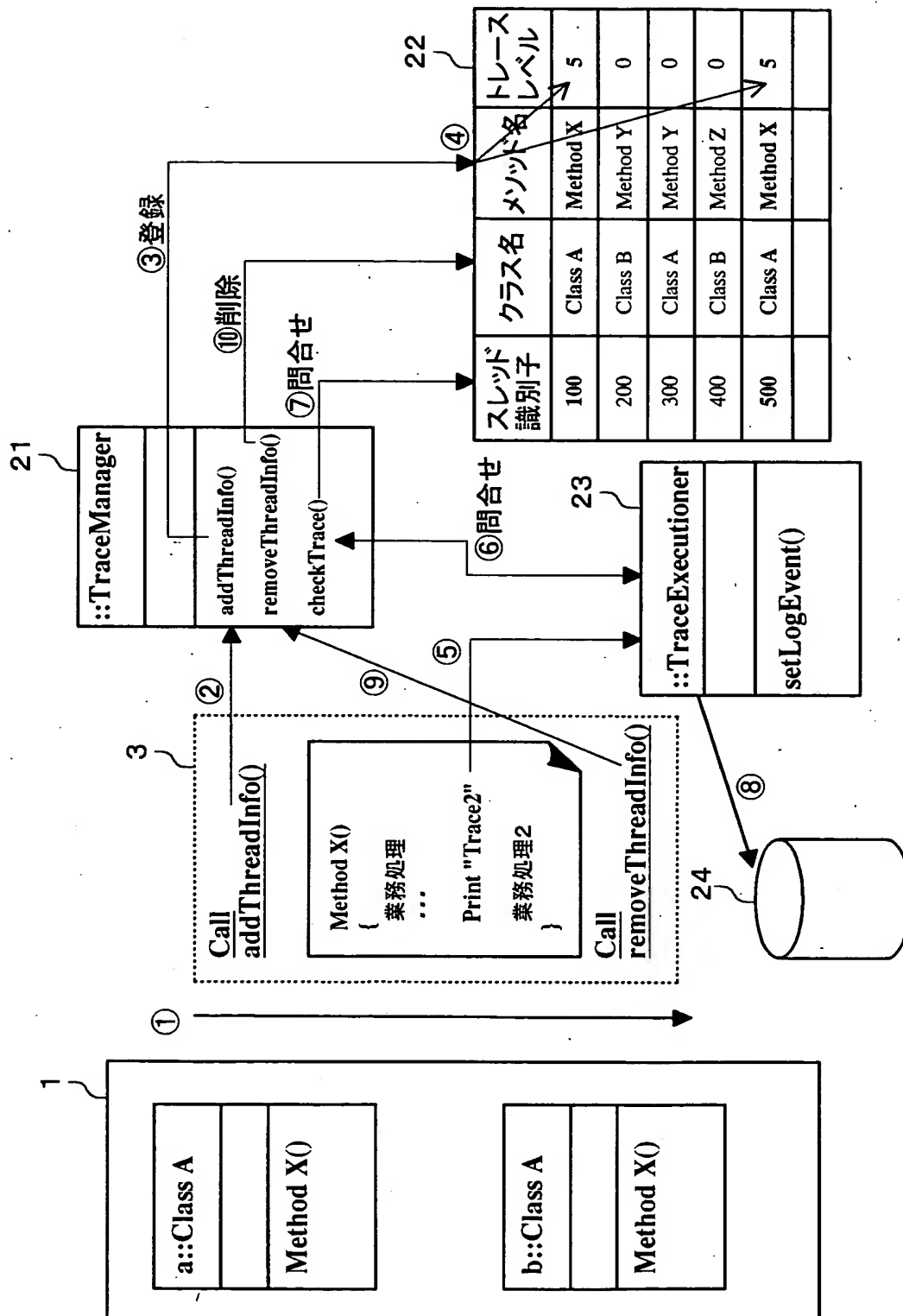
【書類名】

図面

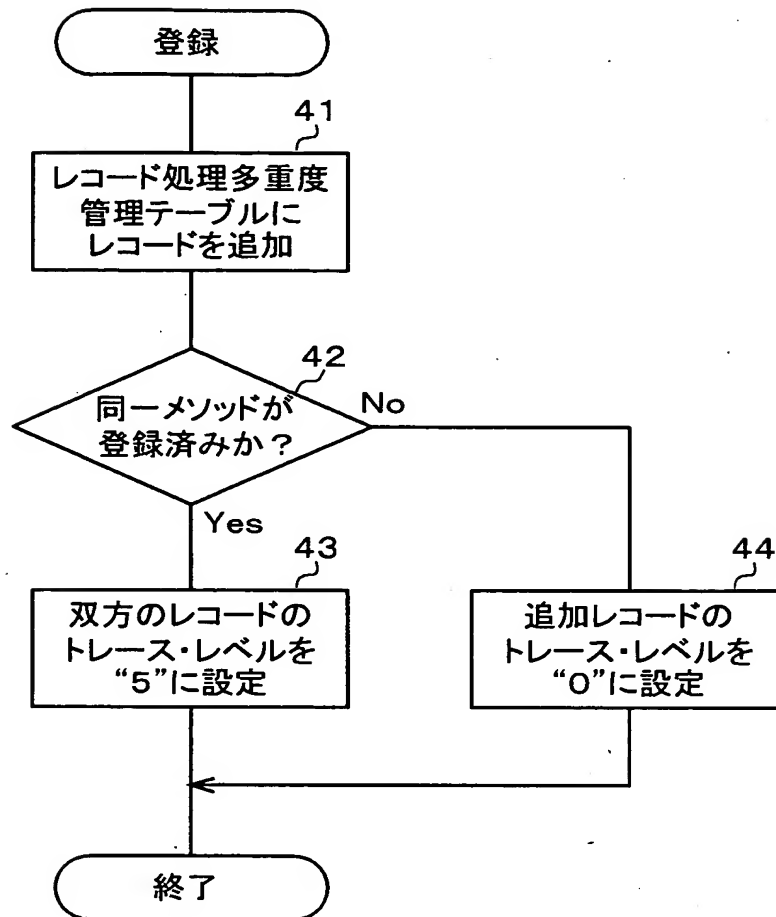
【図 1】



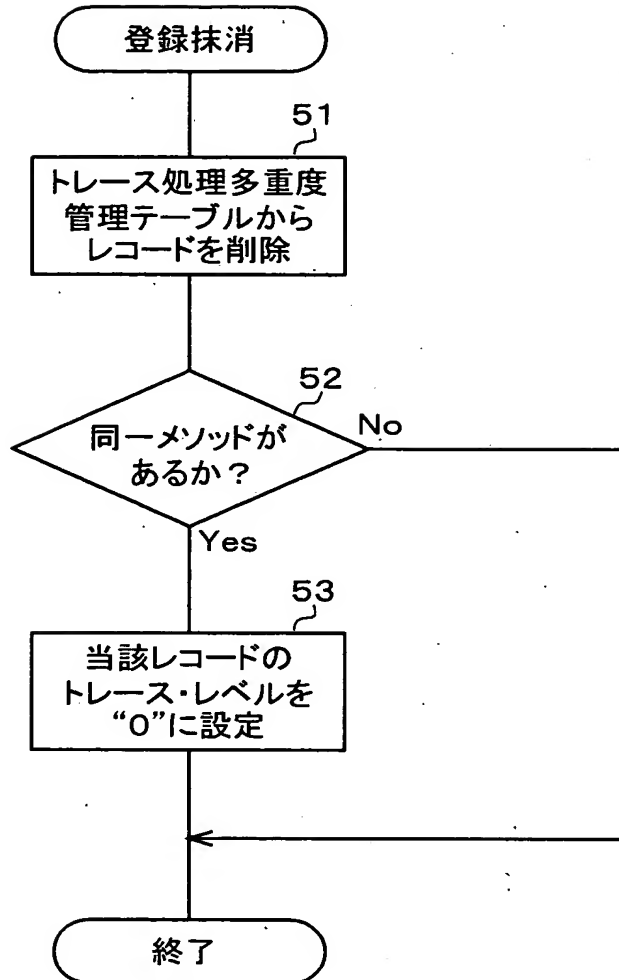
【図 2】



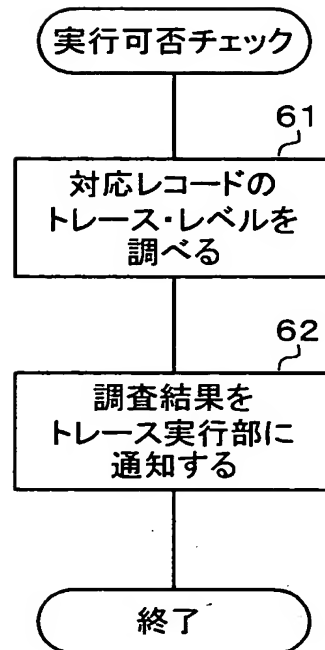
【図3】



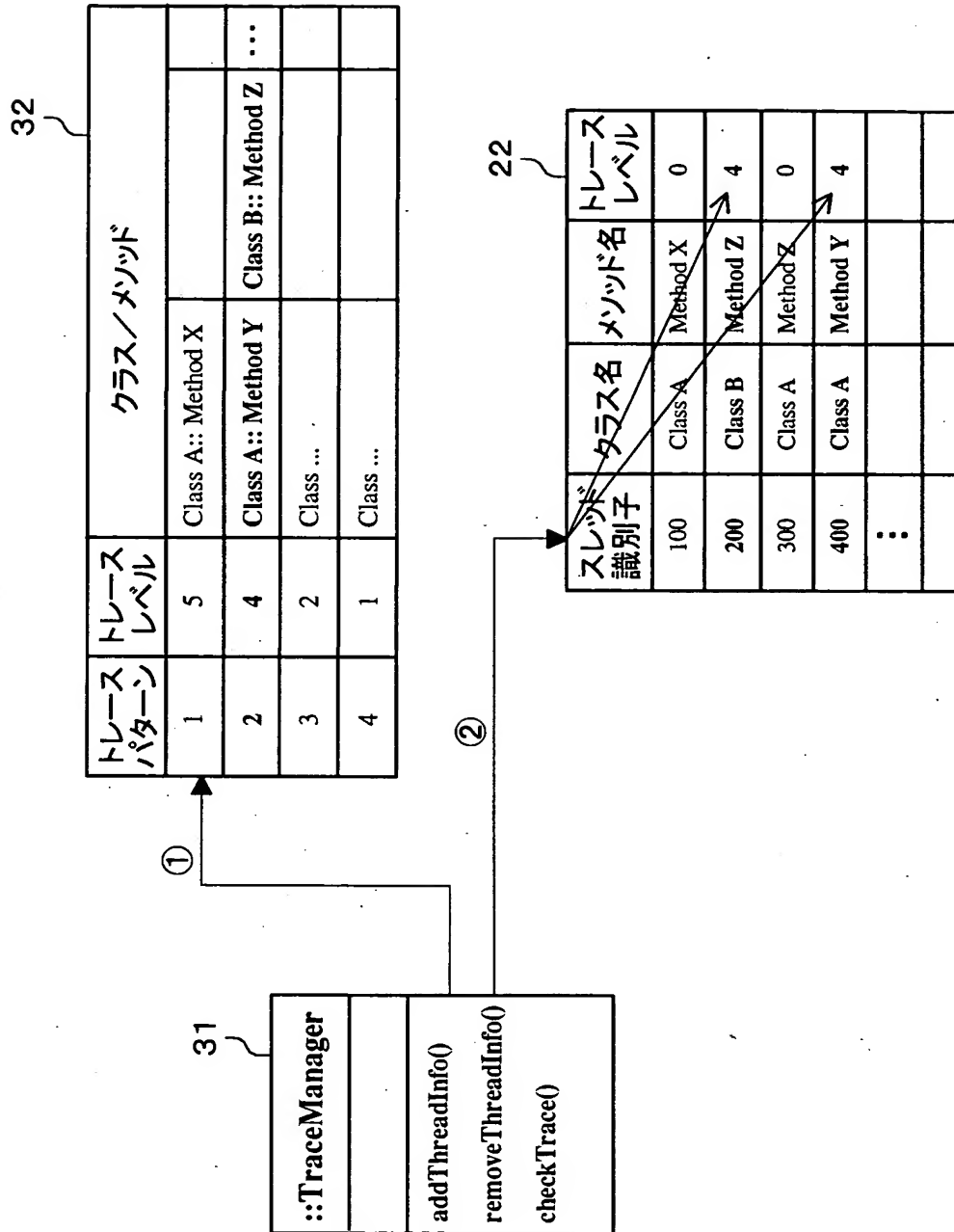
【図 4】



【図 5】

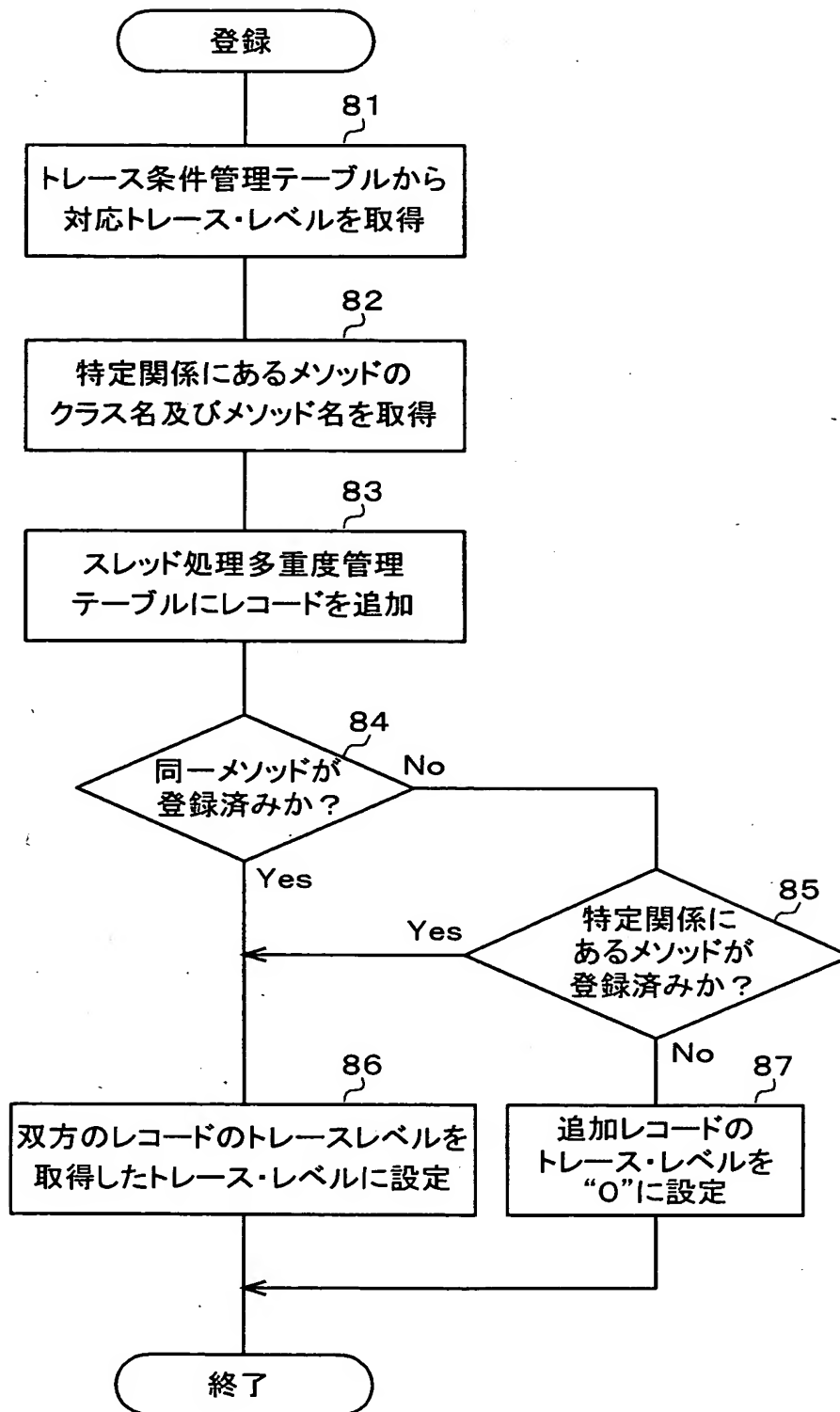


【図 6】

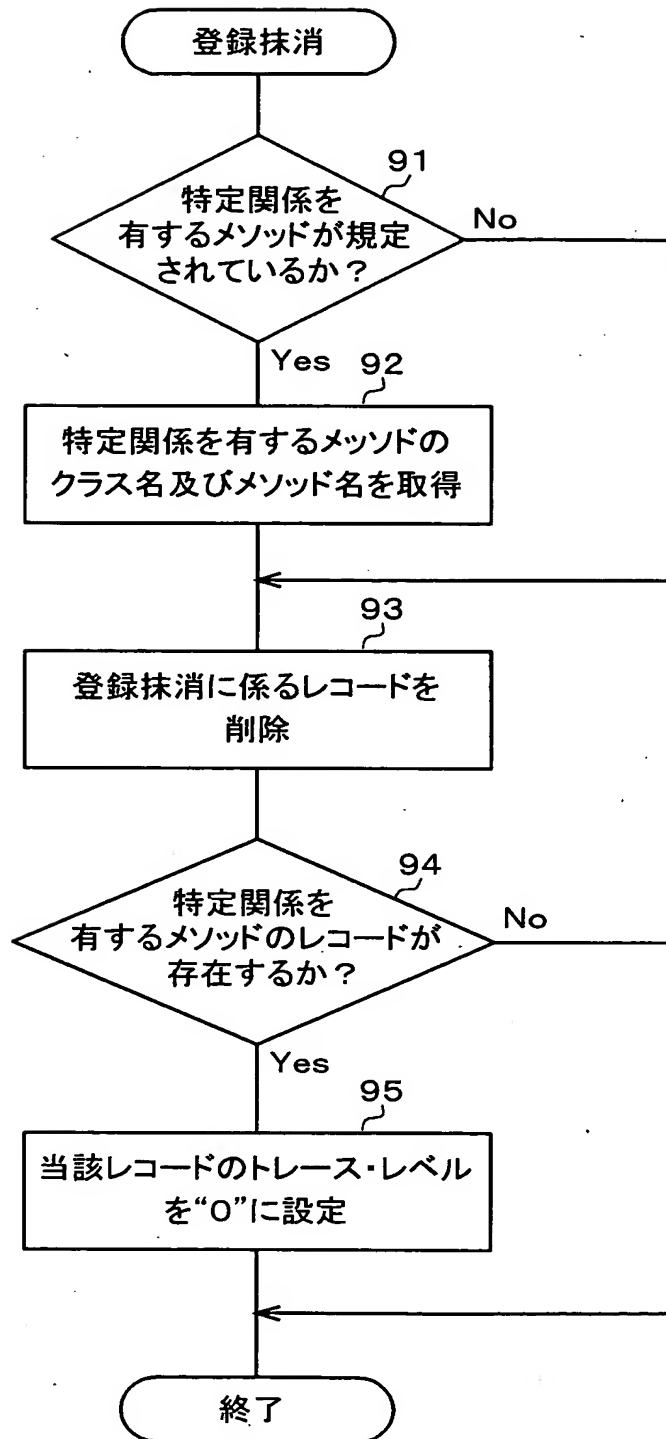




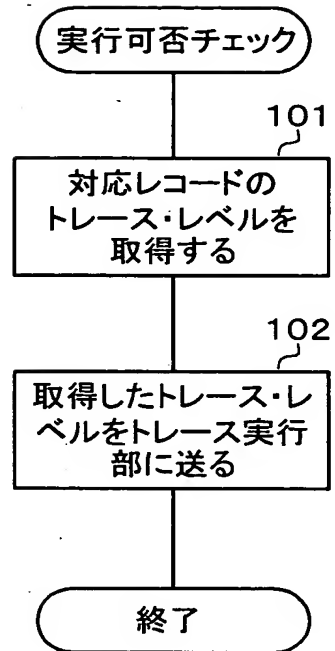
【図 7】



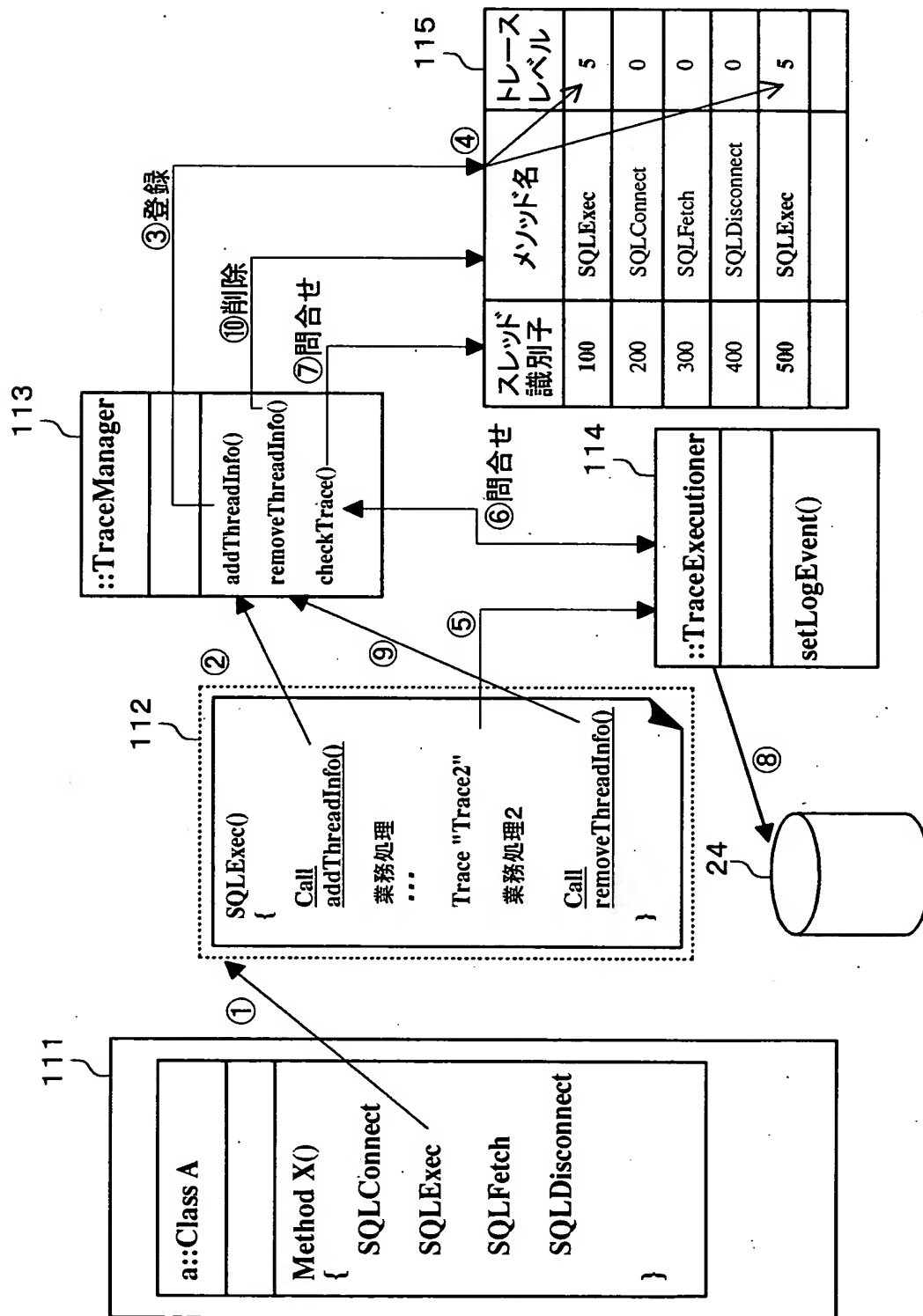
【図 8】



【図 9】



【図 10】



【書類名】 要約書

【要約】

【課題】 プログラムの実行環境に大きな影響を与えることなく、共有リソースを同時にアクセスすることによって生じる問題の解決に必要な情報の収集を行うことができるようにする。

【解決手段】 処理状況に関する情報を収集するトレース処理が含まれるルーチンをマルチスレッドで実行可能な情報処理装置において、起動中のスレッド毎に実行中のルーチンの登録を行い（ステップ 3 1 ～ 3 3、3 5）、実行中のルーチンにおけるトレース処理のレベルを登録情報に基づいて判定する（ステップ 3 3、3 6）。実行中のルーチンにおけるトレース処理のレベル判定は、そのルーチンと同一又は所定の関係にあるルーチンの登録がなされているか否かに基づいて行うことができる。

【選択図】 図 3

認定・付加情報

特許出願の番号	特願2002-366345
受付番号	50201915963
書類名	特許願
担当官	末武 実 1912
作成日	平成15年 3月24日

<認定情報・付加情報>

【提出日】	平成14年12月18日
【特許出願人】	
【識別番号】	390009531
【住所又は居所】	アメリカ合衆国10504、ニューヨーク州 アーモンク ニュー オーチャード ロード
【氏名又は名称】	インターナショナル・ビジネス・マシーンズ・コーポレーション
【代理人】	
【識別番号】	100086243
【住所又は居所】	神奈川県大和市下鶴間1623番地14 日本アイ・ビー・エム株式会社 大和事業所内
【氏名又は名称】	坂口 博
【代理人】	
【識別番号】	100091568
【住所又は居所】	神奈川県大和市下鶴間1623番地14 日本アイ・ビー・エム株式会社 大和事業所内
【氏名又は名称】	市位 嘉宏
【代理人】	
【識別番号】	100108501
【住所又は居所】	神奈川県大和市下鶴間1623番14 日本アイ・ビー・エム株式会社 知的所有権
【氏名又は名称】	上野 剛史
【復代理人】	申請人
【識別番号】	100085408
【住所又は居所】	東京都中央区日本橋2丁目1番1号 櫻正宗ビル 9階
【氏名又は名称】	山崎 隆

次頁無

出 願 人 履 歴 情 報

識別番号 [390009531]

1. 変更年月日 2002年 6月 3日

[変更理由] 住所変更

住 所 アメリカ合衆国10504、ニューヨーク州 アーモンク ニ  
ュー オーチャード ロード

氏 名 インターナショナル・ビジネス・マシーンズ・コーポレーショ  
ン